E-Commerce Management System - Documentation

# 1. Project Overview

The E-Commerce Management System is a web application built using Laravel 12, Laravel Breeze, and Spatie Roles & Permissions. It allows customers to browse products, add them to their cart, and make purchases. Admins and vendors can manage products, track stock, and view reports. The system also supports role-based access, order tracking, and real-time notifications.

# 2. Objectives

- Provide a user-friendly online shopping experience.  
- Automate stock management and purchase logging.  
- Offer a secure payment and checkout process.  
- Enable admins to monitor sales, manage vendors, and track user activity.  
- Create a scalable system ready for future mobile app/API integration.

# 3. Key Features

## Authentication & Roles

- User registration & login (Laravel Breeze)

- Role-based access control (Admin, Vendor, Customer)

- Permissions using Spatie package

## Product Management

- Add, update, delete products

- Categories, subcategories, and brands

- Multiple product images

- Variants (size, color, model)

- Stock alerts (low stock notifications)

## Cart & Orders

- Add to cart, wishlist, save for later

- Order statuses: Pending, Processing, Shipped, Delivered, Completed

- Auto stock update via Laravel Observers

- Invoice/PDF generation for completed orders

- Refunds and return requests

## Payments

- Cash on Delivery

- Online payment gateways (Stripe, PayPal, JazzCash, EasyPaisa)

- Wallet system for users

## Admin Dashboard

- Revenue & sales charts (ChartJS/Recharts)

- Top products, top vendors, customer insights

- Export reports (Excel, CSV, PDF)

- Real-time order tracking

## Customer Panel

- Order history

- Download invoices

- Manage addresses & profile

- Product reviews & ratings

## Vendor Panel

- Upload/manage products

- Track vendor-specific orders

- View vendor sales report

## Notifications & Logs

- Email/SMS notifications for purchases & status updates

- Real-time push notifications (Pusher/WebSockets)

- Activity logs (user purchases, product updates)

## API Support

- REST API for products, orders, cart, wishlist, and authentication

# 4. Database Design

Main Tables:  
- users – customer/vendor/admin records  
- roles & permissions – Spatie for RBAC  
- products – id, name, description, price, stock, vendor\_id  
- categories, subcategories, brands  
- carts – user\_id, product\_id, status (pending/completed)  
- orders – user\_id, total\_price, status  
- order\_items – order\_id, product\_id, qty, price  
- wishlists – user\_id, product\_id  
- reviews – user\_id, product\_id, rating, comment  
- transactions – order\_id, amount, method, status  
- logs – activity tracking

# 5. System Roles

- Admin → Full control, manage products, users, vendors, reports  
- Vendor → Manage their own products and sales  
- Customer → Browse, purchase, review products  
- Delivery Staff (optional) → Update delivery statuses

# 6. Development Roadmap

## Phase 1: Core

- Auth, Roles & Permissions

- Product CRUD + Categories

- Cart & Checkout

- Order & Invoice system

## Phase 2: Enhancements

- Payment gateway integration

- Coupons, discounts, wishlist

- Reviews & ratings

- Vendor panel

## Phase 3: Admin Panel

- Dashboard with sales reports & charts

- Stock alerts

- User/vendor management

- Export reports

## Phase 4: Advanced

- Multi-currency & multi-language

- Delivery management system

- Notifications (email, SMS, push)

- REST API for mobile app

## Phase 5: Enterprise

- Loyalty points & wallet system

- Subscription-based products

- AI recommendations

- Live chat (customer ↔ vendor, vendor ↔ admin)

# 7. Tech Stack

- Backend: Laravel 12  
- Frontend: Blade (Laravel Breeze)  
- Database: MySQL  
- Roles & Permissions: Spatie  
- Payments: Stripe, PayPal, JazzCash, EasyPaisa  
- Reports: DOMPDF, Laravel Excel  
- Realtime: Laravel Echo + Pusher  
- Queues: Redis / Database Queue

# 8. Future Scope

- Convert into a multi-vendor marketplace (like Daraz/Amazon)  
- Develop mobile app (Flutter/React Native) using REST API  
- Add AI-powered recommendations  
- Support multi-language & multi-currency  
- Implement subscription models